Package 'GiANT'

May 7, 2026

Type Package

Title Gene Set Uncertainty in Enrichment Analysis

Version 1.3.4

Date 2024-09-02

Maintainer Hans A. Kestler <hans.kestler@uni-ulm.de>

Description Toolbox for various enrichment analysis methods and quantification of uncertainty of gene sets, Schmid et al. (2016) <doi:10.1093/bioinformatics/btw030>.

License Artistic-2.0

LazyLoad yes

Depends R (>= 3.5.0)

Imports parallel, graphics, grDevices, methods, stats, utils

Suggests fdrtool, st, limma, globaltest, DESeq2, GlobalAncova

Encoding UTF-8

NeedsCompilation no

Author Florian Schmid [aut],
Christoph Muessel [aut],
Johann M. Kraus [aut],
Hans A. Kestler [aut],
Hans A. Kestler [cre]

Repository CRAN

Date/Publication 2024-09-05 16:50:06 UTC

Contents

`GiANT-package` *Enrichment analysis*

Description

Toolbox for gene set analysis of uncertain gene sets.

Details

Package:	GiANT
Type:	Package
Version:	1.3
Date:	2020-04-29
License:	Artistic-2.0
LazyLoad:	yes

This package provides an approach for evaluating the fuzziness of a gene set. This is done by repeatedly performing gene set analyses on slightly modified versions of the gene set and comparing their enrichment scores. A utility for such uncertainty tests is provided in the `evaluateGeneSetUncertainty` function.

The package also comprises a generic framework for different types of enrichment analyses (Ackermann and Strimmer). It establishes a customizeable pipeline that typically consists of the following steps:

- Calculation of gene-level statistics: A gene-level statistic scores the relationship between the measurements for a specific gene and the class labels. Typical measures include correlation coefficients, the t statistic or the fold change between the groups (see `gls` for gene-level statistics included in the package).

- Transformation of gene-level statistic values: Optionally, the gene-level statistic values can be postprocessed, e.g. by taking the absolute value or the square for correlation values or by binarizing or ranking values.
  See `transformation` for transformations included in the package.
- Calculation of gene set statistics: Based on the (possibly transformed) gene-level statistics, the gene set(s) of interest is/are scored. Examples are the median, the mean or the enrichment score of the gene-level statistic values in the gene set(s). See `gss` for gene set statistics included in the package.
- Significance assessment: To assess the significance of the gene set statistic value(s) with respect to a null distribution, computer-intensive tests are performed. These tests repeatedly sample random label vectors or gene sets and calculate their gene set statistic values. These values can then be compared to the true gene set statistics. See `significance` for significance assessment methods included in the package.

The package represents such analysis pipelines as configuration objects that can be created using the function `gsAnalysis`. For predefined state-of-the-art methods, such as Gene Set Enrichment Analysis (Subramanian et al), Overrepresentation Analysis or Global Ancova (Hummel et al), it provides predefined configurations (see `predefinedAnalyses`).

The main function for standard gene set analyses, `geneSetAnalysis`, performs enrichment analyses based on pipeline configuration objects.

Author(s)

Florian Schmid, Christoph Müssel, Johann M. Kraus, Hans A. Kestler

Maintainer: Hans A. Kestler <hans.kestler@uni-ulm.de>

References

Ackermann, M., Strimmer, K. (2009) A general modular framework for gene set enrichment analysis. *BMC Bioinformatics*, **10**(1), 47.

Subramanian, A., Tamayo, P., Mootha, V. K., Mukherjee, S., Ebert, B. L., Gillette, M. A., Paulovich, A., Pomeroy, S. L., Golub, T. R., Lander, E. S., Mesirov, J. P. (2005) Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Science of the United States of America*, **102**, 15545-15550.

Hummel, M., Meister, R., Mansmann, U. (2008) Globalancova: exploration and assessment of gene group effects. *Bioinformatics*, **24**(1), 78–85.

Examples

```
data(exampleData)
################################
# Example 1: gene set analysis   #
################################
res <- geneSetAnalysis(
  # parameters for geneSetAnalysis
  dat = countdata,
  geneSets = pathways[1],
  analysis = analysis.averageCorrelation(),
```

```
  adjustmentMethod = "fdr",
  # additional parameters for analysis.averageCorrelation
  labs = labels,
  method = "pearson",
  numSamples = 50)

summary(res, mode="table")

###################################
# Example 2: uncertainty analysis  #
###################################
resUncertainty <- evaluateGeneSetUncertainty(
  # parameters for evaluateGeneSetUncertainty
  dat = countdata,
  geneSet = pathways[[3]],
  analysis = analysis.averageCorrelation(),
  numSamplesUncertainty = 5,
  blockSize = 1,
  k = seq(0.1,0.9,by=0.1),
  # additional parameters for analysis.averageCorrelation
  labs = labels,
  numSamples = 5)

plot(resUncertainty, main = names(pathways[3]))
```

`countdata` *Example data*

Description

Randomly generated example data representing a count matrix

Usage

```
data(exampleData)
```

Details

The data consists of a variable `countdata` with a matrix with 1113 genes and 96 samples. The count values are randomly generated numbers.

Examples

```
data(exampleData)

print(countdata)
```

createSummaryTable *Create an overview table for an analysis*

Description

Creates a data frame summarizing an analysis. This table has one row per gene set, each comprising the adjusted and unadjusted p-values and the number of genes for the set.

Usage

```
createSummaryTable(object,
orderBy = c("adjustedPValues", "rawPValues", "geneSetName"),
significantOnly = FALSE,
signLevel = object$signLevel)
```

Arguments

object	A result object as returned by geneSetAnalysis.
orderBy	Specifies which field should be used for the row ordering. By default, rows are ordered according to the adjusted p-values.
significantOnly	Specifies whether all gene sets (significantOnly=FALSE) or only the statistically significant gene sets (significantOnly=TRUE) should be included in the table.
signLevel	If significantOnly=TRUE, this specifies the significance level for the results that should be included in the table. By default, the original significance level of the analysis is used.

Value

A data frame with one row for each included gene set and the columns "adjustedPValues", "rawPValues", "geneSetName" and "geneSetSize". For overrepresentation analyses, there is an additional column "intersectSize" specifying the size of the intersection of the core set and the corresponding gene set.

See Also

geneSetAnalysis, hist.gsaResult, summary

Examples

```
# load data
data(exampleData)
  # perform gene set analyses for several pathways
  res <- geneSetAnalysis(
   # global parameters
  dat = countdata,
```

```
  geneSets = pathways,
  analysis = analysis.averageCorrelation(),
  # additional parameters for analysis.averageCorrelation
  labs = labels,
  numSamples = 10)

  tab <- createSummaryTable(res)
```

`evaluateGeneSetUncertainty` *Quantify gene set uncertainty*

Description

A robustness measure that quantifies the uncertainty of a gene set by performing a resampling experiment and can be used in the `robustness` parameter of `gsAnalysis`.

Usage

```
evaluateGeneSetUncertainty(
...,
dat,
geneSet,
analysis,
numSamplesUncertainty,
blockSize = 1,
k = seq(0.01, 0.99, by=0.01),
signLevel = 0.05,
preprocessGeneSet = FALSE,
cluster = NULL)
```

Arguments

`...`	Additional parameters for the different steps of the analysis pipeline, depending on the concrete configuration supplied in `analysis`.
`dat`	A numeric matrix of gene expression values for all analyzed genes. Here, each row corresponds to one gene, and each column corresponds to one sample. The rows must be named with the gene names used in the gene sets.
`geneSet`	A vector containing the names of genes in a gene set. All genes set must correspond to the row names of `dat`.
`analysis`	The parameters of the analysis that is applied to the perturbed copies of the gene set. These parameters are described by an object of class `gsAnalysis` as returned by the function `gsAnalysis` or the predefined analysis descriptors in `predefinedAnalyses`.
`numSamplesUncertainty`	The number of resampling experiments which should be applied to estimate the robustness of `geneSet`.

`blockSize` Number of genes in one resampled block.

`k` A `vector` of percentages of genes in the randomized gene sets that should be taken from the original gene set. The remaining genes are chosen randomly. For each value a resampling experiment is performed.

`signLevel` The significance level for the significance assessment of the gene sets (defaults to `0.05`).

`preprocessGeneSet`
Specifies whether the gene sets in `geneSets` should be preprocessed or not. If set to `TRUE`, all genes that are not part of the data set (i.e. not in `rownames(dat)`) are removed from the gene sets.

`cluster` If the analyses should be applied in parallel for the different values of `k`, this parameter must hold an initialized cluster as returned by `makeCluster`. If this parameter is `NULL`, the analyses are performed sequentially.

Details

The uncertainty analysis repeatedly replaces parts of the original gene sets by random genes and calculating the gene set statistics for these randomized gene sets. This yields a distribution of gene set statistic values for slightly modified variants of the original gene set.

Value

Returns a list (of class `uncertaintyResult`) with the following elements:

`uncertainty` The calculated stability of the original gene set.

`confidenceValues`
A matrix of quantiles of `gssValues` (signLevel, 0.5, 1-signLevel). One row for each value in `k`.

`uncertaintyEvaluations`
A list with one entry per value in `k` containing the following elements:

- Quantiles of `gssValues`: signLevel, 0.5, 1-signLevel.
- gssValues: A vector of gene set statistic values, one for each randomly sampled gene set.
- uncertainGeneSets: A matrix containing all partially random gene sets.
- k: The percentage of genes in the randomized gene sets taken from the original gene set.

`signLevel` The significance level used for this analysis.

`originalGeneSetValues`
Result of `geneSetAnalysis` for the original `geneSet`.

See Also

`geneSetAnalysis`, `gsAnalysis`, `gls`, `transformation`, `gss`, `plot.uncertaintyResult`

Examples

```
data(exampleData)

  res <- evaluateGeneSetUncertainty(
   # parameters for evaluateGeneSetUncertainty
   dat = countdata,
   geneSet = pathways[[1]],
   analysis = analysis.averageCorrelation(),
   numSamplesUncertainty = 10,
   k = seq(0.1,0.9, by=0.1),
   # additional parameters for analysis.averageCorrelation
   labs = labels,
   numSamples = 10)
```

Filter gene sets *Filtering of gene sets*

Description

Filters gene sets according to different criteria.

Usage

```
filterGeneSets(
geneSets,
includedGenes = NULL,
minIntersectSize = length(includedGenes),
adjMatrix = NULL,
steps = NULL)
```

Arguments

`geneSets`	A list of gene sets, where each gene set is a vector of gene names corresponding to the row names of `dat`.
`includedGenes`	A vector of gene names whose occurence in each of the gene sets is checked. The further parameters how these genes are used to filter gene sets
`minIntersectSize`	If this parameter is not `NULL`, only gene sets with an intersection of at least `minIntersectSize` genes with respect to `includedGenes` (or `includedGenes` expanded by its interactions if `adjMatrix` and `steps` are supplied) are included in the result set. By default, this is the size of `includedGenes`, requiring `includedGenes` to be a subset of each gene set.
`adjMatrix`	An optional adjacency matrix in which an entry is `1` if there is a direct interaction between the corresponding genes and `0` otherwise. If this is non-null, the set of genes in `includedGenes` is expanded by adding all genes whose distance in the adjacency graph is at most `steps`.

`steps` The maximum distance of interacting genes to the genes in `includedGenes` according to `adjMatrix` to be added to the expanded gene list. E.g., `steps = 1` means that all genes which are direct interaction partners of the initial genes in `includedGenes` are added to `includedGenes`.

Value

Returns a filtered list of gene sets with the same structure as `geneSets`.

See Also

`geneSetAnalysis`, `preprocessGs`

Examples

```
geneSets <- list(
gs1 = paste("gene",1:20,sep=""),
gs2 = paste("gene",50:60,sep=""),
gs3 = paste("gene",90:92,sep=""),
gs4 = paste("gene",55:65,sep="")
)

newGeneSets1 <- filterGeneSets(
geneSets = geneSets,
includedGenes = c("gene55","gene60"))

newGeneSets2 <- filterGeneSets(
geneSets = geneSets,
includedGenes = c("gene1","gene55","gene20","gene100"),
minIntersectSize = 2)

examplePathway <- c("gene1","gene2","gene3","gene4")
pathwayAdjMatrix <- matrix(0,100,100)
rownames(pathwayAdjMatrix) <- paste("gene",1:100,sep="")
colnames(pathwayAdjMatrix) <- paste("gene",1:100,sep="")

# gene1 interacts with gene2 and gene3
# -> step 1 if gene1 is starting point
pathwayAdjMatrix[1,2:3] <- 1
pathwayAdjMatrix[2:3,1] <- 1

# gene3 interacts with gene4
# -> step 2 if gene1 is starting point
pathwayAdjMatrix[3,4] <- 1
pathwayAdjMatrix[4,3] <- 1

newGeneSets3 <- filterGeneSets(
geneSets = geneSets,
includedGenes = "gene1",
adjMatrix = pathwayAdjMatrix,
steps = 2)
```

`Gene set analysis` *Main interface for enrichment analyses.*

Description

The main function of the package that performs a gene set analysis for a list of gene sets.

Usage

```
geneSetAnalysis(
...,
dat,
geneSets,
analysis,
signLevel = 0.05,
preprocessGeneSets = FALSE,
adjustmentMethod = p.adjust.methods,
cluster = NULL)
```

Arguments

`...`	Additional parameters for the different steps of the analysis pipeline, depending on the concrete configuration supplied in `analysis`.
`dat`	A numeric matrix of gene expression values for all analyzed genes. Here, each row corresponds to one gene, and each column corresponds to one sample. The rows must be named with the gene names used in the gene sets.
`geneSets`	A list of gene sets, where each gene set is a vector of gene names corresponding to the row names of `dat`.
`analysis`	An object of type `gsAnalysis` as returned by `gsAnalysis` or by the predefined configurations (see `predefinedAnalyses`).
`signLevel`	The significance level for the significance assessment of the gene sets (defaults to `0.05`).
`preprocessGeneSets`	Specifies whether the gene sets in `geneSets` should be preprocessed or not. If set to `TRUE`, all genes that are not part of the data set (i.e. not in `rownames(dat)`) are removed from the gene sets.
`adjustmentMethod`	The method to use for the adjustment for multiple testing (see `method` parameter of `p.adjust` for possible values).
`cluster`	If the analyses should be applied in parallel for the gene sets, this parameter must hold an initialized cluster as returned by `makeCluster`. If this parameter is `NULL`, the analyses are performed sequentially.

Details

This is the main interface function of the package for gene set enrichment analyses. Analyses usually consist of a pipeline of steps. Often, the first step is the calculation of a summary statistic for the relation of each gene to the class labels. These values or transformations thereof are employed to calculate a gene set statistic for each of the supplied gene sets. The significance of gene set enrichments can be determined according to different methods, and the robustness of gene sets can be evaluated by slightly modifying the gene sets. To provide a flexible mechanism for the plethora of different approaches arising from the different choices, basic pipeline configurations are encapsulated in `gsAnalysis` objects which can be created using the `gsAnalysis` function. Ready-to-use configuration objects for certain well-known methods are included in the package (see `predefinedAnalyses`). Parameters of the chosen analysis pipeline can be set in the `...` parameter.

Value

An object of the type `gsaResult` with the following elements:

`adjustedPValues`
: A vector of p-values, one for each gene set. These values are already adjusted for multiple testing according to the `adjustmentMethod` parameter.

`rawPValues`
: The raw unadjusted p-values, one for each gene set.

`res.all`
: A list comprising the detailed results for each gene set. Each element of this list is another list with the following components:

  `pValue`: The raw (unadjusted) p-value for the gene set.

  `geneSetValues`: If `analysis` is a global analysis, this is the object returned by the method for the corresponding gene set. For an analysis pipeline, this holds the values of the gene-level statistic, the transformed values and the values of the gene set statistic (see also `gsAnalysis`).

  `significanceValues`: Gene set statistics for each randomly drawn gene set for significance assessment and a `list` of this gene sets. Only set for analysis of type 'geneSetAnalysis'. `NULL` for 'global' analysis.

  `geneSet`: The supplied gene set.

`signLevel`
: The significance level used for this analysis.

`analysis`
: The performed analysis (of type `gsAnalysis`).

`analysisType`
: A character string identifying the analysis as an enrichment analysis pipeline (`"geneSetAnalysis"`) or as a global analysis (`"global"`).

`adjustmentMethod`
: The method used to adjust the p-values in `adjustedPValues`

References

Ackermann, M., Strimmer, K. (2009) A general modular framework for gene set enrichment analysis. *BMC Bioinformatics*, **10**(1), 47.

See Also

`gsAnalysis`, `gls`, `transformation`, `gss`, `global`, `significance`, `evaluateGeneSetUncertainty`, `hist.gsaResult`, `preprocessGs`

Examples

```
# load data
data(exampleData)

  # apply predefined analysis for gene set enrichment analysis
  res <- geneSetAnalysis(
   # parameters for geneSetAnalysis
   dat = countdata,
   geneSets = pathways[1],
   analysis = analysis.averageCorrelation(),
   adjustmentMethod = "fdr",
   # additional parameters for analysis.averageCorrelation
   labs = labels,
   method = "pearson",
   numSamples = 10)
```

GeneLevelStatistics *Gene-level statistics*

Description

Functions to calculate the gene-level statistic, as used in the `gls` parameter of `gsAnalysis`. A gene-level statistic calculates a measure of correlation between the expression of a gene and the class labels.

Usage

```
gls.cor(dat, labs, method = "pearson")

gls.regression(dat, labs)

gls.foldChange(dat, labs, logMeasurements = TRUE)

gls.tStatistic(dat, labs, pValue = FALSE, alternative = "two.sided")

gls.moderateTStatistic(dat,labs)

gls.nBinomTest(dat, labs,
returnValue = c("pval", "qval", "foldChange", "log2FoldChange"),
dispersionMethod = "blind",
dispersionSharingMode = "fit-only",
dispersionFitType = "local")
```

Arguments

`dat`	A numeric matrix of gene expression values for all analyzed genes. Here, each row corresponds to one gene, and each column corresponds to one sample. The rows must be named with the gene names used in the gene sets.

`labs` A vector of class labels for the samples in `dat`.

`logMeasurements`
For `gls.foldChange`, whether the values in `dat` are logarithmized (`logMeasurements=TRUE`) or not (`logMeasurements=FALSE`).

`method` For `gls.cor`, the correlation method to be used (see `cor`).

`pValue` For `gls.tStatistic`, this specifies whether the p-value (`pValue=TRUE`) or the test statistic (`pValue=FALSE`) of the t test should be returned.

`alternative` For `gls.tStatistic`, this specifies the alternative of the t-test. See also `t.test`.

`returnValue` For `gls.nBinomTest`, this determines the type of values values that should be returned. `"pval"` returns the raw p-values, `"qval"` returns the p-values adjusted by the FDR, `"foldChange"` returns the fold changes, and `"log2FoldChange"` returns the log2 fold changes. For more details, see `results`.

`dispersionMethod`
For `gls.nBinomTest`, this specifies how the empirical dispersion is computed (see `estimateDispersions`).

`dispersionSharingMode`
For `gls.nBinomTest`, this specifies which values should be used by `results` (fitted values or empirical values, see `estimateDispersions` for more details).

`dispersionFitType`
For `gls.nBinomTest`, this determines the method for fitting the dispersion-mean relation (see `estimateDispersions`).

Details

Standard functions for the calculation of gene-level statistics (to be used in an analysis pipeline defined by `gsAnalysis`):

- `gls.cor`: Calculates the correlation of the gene expression values to the class labels.
- `gls.regression`: Calculates the slope of a linear regression of the gene expression values and the class labels.
- `gls.foldChange`: Calculates the (standard or log2) fold change between the measurements for the two classes.
- `gls.tStatistic`: Calculates the p-value or the statistic of a two-sample t test for the measurements of the two classes
- `gls.moderateTStatistic`: Calculates the moderate t statistic for the measurements of the two classes
- `gls.nBinomTest`: Applies the negative binomial test for sequencing data based on the **DESeq2** package to test for differences between two classes (see `results`).

Value

Each of these function returns a numeric vector of gene-level statistics (one entry per gene).

See Also

`geneSetAnalysis`, `gsAnalysis`, `gss`, `transformation`

GeneSetStatistics *Gene set statistics*

Description

Functions to calculate a gene set statistic, as used in the gss parameter of gsAnalysis. A gene set statistic summarizes a single gene set.

Usage

```
gss.mean(x, geneSetIndices)

gss.sum(x, geneSetIndices)

gss.wilcoxonRankTest(x, geneSetIndices)

gss.maxmean(x, geneSetIndices)

gss.median(x, geneSetIndices)

gss.enrichmentScore(x, geneSetIndices, p = 1)

gss.fisherExactTest(x, geneSetIndices)

gss.gsz(x, geneSetIndices, w1 = 0.2, w2 = 0.5, preVar = 0, varConstant = 10)
```

Arguments

x	A vector comprising one numeric value for each gene in the data set. This vector is usually obtained from the previous step, the gene-level statistic (see gls) or the transformed gene-level statistic (see transformation).
geneSetIndices	A vector containing the indices of the genes in the gene set with respect to the full gene set (i.e., the indices of the rows containing the measurements for these genes in dat).
p	Factor for gss.enrichmentScore that specifies the way hits are weighted. For p = 0, the enrichment score is a Kolmogorov-Smirnov statistic. For p = 1 (the default), hits are weighted by their correlation.
w1	Weight for the median of the variance estimates for a gene set of size varConstant. Should be between 0 and 1. Default is w1 = 0.2.
w2	Weight for the median of the variance estimates for a gene set across the whole gene list. Should be between 0 and 1. Default is w2 = 0.5.
preVar	Parameter for incorporating the uncertainty of the observations. This is omitted by default (preVar = 0).
varConstant	Reference gene set size used for variance estimates. Default is varConstant = 10.

Details

Standard functions for the calculation of gene set statistics (to be used in an analysis pipeline defined by `gsAnalysis`):

- `gss.mean`: Calculates the mean of the (transformed) gene-level statistic values for the genes in the set.
- `gss.sum`: Calculates the sum of the (transformed) gene-level statistic values for the genes in the set.
- `gss.wilcoxonRankTest`: Calculates a Wilcoxon test comparing the (transformed) gene-level statistic values for the genes in the set versus those of the genes not in the set.
- `gss.maxmean`: Calculates the maximum of the means of positive and negative statistic values, weighted by the overall proportion of positive/negative values (e.g. for correlation scores where the sign denotes the direction). Described in Efron and Tibshirani.
- `gss.median`: Calculates the median of the (transformed) gene-level statistic values for the genes in the set.
- `gss.enrichmentScore`: Calculates the enrichment score of the (transformed) gene-level statistic values for the genes in the set, as described in Subramanian et al.
- `gss.fisherExactTest`: Performs Fisher's exact test to check gene sets for overrepresentation in the differential genes. This should be used used in combination with the transformation `transformation.adjustAndBinarize`.
- `gss.gsz`: Calculates the Gene Set Z-score of the (transformed) gene-level statistic values for the genes in the set. Described in Toronen et al.

Value

Each method returns a single numeric value, the gene set statistic for the supplied gene set.

References

Efron, B., Tibshirani, R. (2007) On testing the significance of sets of genes. *Annals of Applied Statistics*, **1**, 107-129.

Subramanian, A., Tamayo, P., Mootha, V. K., Mukherjee, S., Ebert, B. L., Gillette, M. A., Paulovich, A., Pomeroy, S. L., Golub, T. R., Lander, E. S., Mesirov, J. P. (2005) Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Science of the United States of America*, **102**, 15545–15550.

Toronen, P., Ojala, P. J., Marttinen, P., Holm L. (2009) Robust extraction of functional signals from gene set analysis using a generalized threshold free scoring function. *BMC Bioinformatics*, **10**(1), 307.

See Also

`geneSetAnalysis`, `gsAnalysis`, `gls`, `transformation`

GlobalAnalysis *Global analyses*

Description

Functions to perform global gene set analyses, as used in the globalStat parameter of gsAnalysis.

Usage

```
global.overrepresentation(dat,
geneSet,
coreSet)

global.ancova(dat,
geneSet,
labs,
...)

global.test(dat,
geneSet,
labs,
...)
```

Arguments

dat	A numeric matrix of gene expression values for all analyzed genes. Here, each row corresponds to one gene, and each column corresponds to one sample. The rows must be named with the gene names used in the gene sets.
geneSet	A gene set in form of a vector of gene names corresponding to the row names of dat.
coreSet	A gene set of interest resulting from an analysis of dat that should be compared to geneSet in the overrepresentation analysis. This is also a vector of gene names corresponding to the row names of dat.
labs	A vector of class labels for the samples in dat.
...	Further parameters for GlobalAncova and gt, as defined in the corresponding manual pages. The parameters xx, test.genes and group are set automatically by global.ancova, and the parameters alternative, subsets and response are set automatically by global.test.

Details

Wrapper functions for global gene set analyses.

- global.overrepresentation: This function performs an overrepresentation analysis by rating the overlap of geneSet and coreSet with respect to the set of all genes using Fisher's exact test.

- `global.ancova`: This function performs a global gene set enrichment analysis using the global ANCOVA method by Hummel et al. It wraps the `GlobalAncova` function in the **GlobalAncova** package.
- `global.test`: This function performs a global gene set enrichment analysis using a global test by Goeman et al. It wraps the `gt` function in the **globaltest** package.

Value

A list containing the following items:

`pValue`	The p-value for the significance of `geneSet`.
`intersectGeneSetCoreSet`	This element is only returned in case of an overrepresentation analysis and consists of a vector of genes included in both sets (`geneSet` and `coreSet`).
`res.all`	The full result object returned by `fisher.test`, `GlobalAncova` or `gt` respectively.

References

Hummel, M., Meister, R., Mansmann, U. (2008) GlobalANCOVA: exploration and assessment of gene group effects. *Bioinformatics*, **24**(1), 78–85.

Goeman, J. J., van de Geer, S. A., de Kort, F., van Houwelingen, H. C. (2004) A global test for groups of genes: testing association with a clinical outcome. *Bioinformatics*, **20**(1), 93–99.

See Also

`geneSetAnalysis`, `gsAnalysis`

`gsAnalysis` *Gene set analysis.*

Description

Defines the configuration of an analysis that can be performed using `geneSetAnalysis`, and returns it as a wrapper object.

Usage

```
gsAnalysis(name,
gls = NULL,
glsParameterNames = NULL,
transformation = NULL,
transformationParameterNames = NULL,
gss = NULL,
gssParameterNames = NULL,
globalStat = NULL,
globalStatParameterNames = NULL,
```

```
significance = NULL,
significanceParameterNames = NULL,
testAlternative = c("greater", "less"))
```

Arguments

name	A character string describing the analysis.
gls	The name of the function that calculates the gene-level statistic for a given dataset. If set to NULL, it is assumed that the input data already comprises gene-level statistic values, and the input is directly supplied to transformation. The first (fixed) parameter of a gls function is the dataset.
glsParameterNames	A character vector of names of the parameters used by the gene-level statistic defined in gls.
transformation	The name of the function that transforms the gene-level statistics values. If set to NULL, the values supplied by gls are directly handed over to gss. The only fixed parameter for transformation is the gene-level statistic (supplied as first parameter).
transformationParameterNames	A character vector of names of the parameters used by the transformation defined in transformation.
gss	The name of the function that calculates the gene set statistics from untransformed or transformed gene-level statistic values. If set to NULL, the values supplied by transformation are directly handed over to significance. Fixed parameters are the transformed values (first parameter) and geneSetIndices containing the (row-) indices of the current gene set genes in the dataset.
gssParameterNames	A character vector vector of names of the parameters used by the gene set statistic defined in gss.
globalStat	If the gene set analysis consists of a global test and cannot be divided into the calculation of a gene-level statistic, a transformation and the calculation of the gene set statistics, this parameter defines the name of the function that performs this global analysis. In this case, the parameters gls, transformation and gss are ignored. Fixed parameters are dat containing the whole dataset and geneSet containing the current gene set.
globalStatParameterNames	A character vector names of the parameters used by the global analysis defined in globalStat.
significance	The name of a method that performs a significance assessment for the gene set statistic values. If set to NULL, geneSetAnalysis does not return p-values, but returns the statistics supplied by gss or globalStat. Fixed parameters are dat containing the whole dataset, geneSet containing the current gene set, analysis with the supplied gsAnalysis and glsValues with (depending on whether a transformation is supplied or not) transformed gene-level statistics for each gene in the dataset.

`significanceParameterNames`
: A character vector of names of the parameters used by the significance assessment method defined in `significance`.

`testAlternative`
: Specifies the alternative hypothesis of the significance test for the gene set enrichment, which may be dependent on the chosen gene set statistic. Must be one of `"greater"` or `"less"`.

Details

The function provides a way of flexibly defining the steps of the gene set analysis pipeline. This pipeline consists of a subset of the following steps, each of which may have specific parameters:

- Gene-level analysis: A gene-level statistic scores the relationship between the measurements for a specific gene and the class labels. Typical measures include correlation coefficients, the t statistic or the fold change between the groups (see `gls` for gene-level statistics included in the package).
- Transformation of gene-level statistics: Optionally, the gene-level statistic values can be post-processed, e.g. by taking the absolute value or the square for correlation values or by binarizing or ranking values. See `transformation` for transformations included in the package.
- Gene set analysis: Based on the (possibly transformed) gene-level statistics, the gene set(s) of interest is/are scored. Examples are the median, the mean or the enrichment score of the gene-level statistic values in the gene set(s). See `gss` for gene set statistics included in the package.
- Significance assessment: To assess the significance of the gene set statistic value(s) with respect to a null distribution, computer-intensive tests are performed. These tests repeatedly sample random label vectors or gene sets and calculate their gene set statistic values. These values can then be compared to the true gene set statistics. See `significance` for significance assessment methods included in the package.
- Global analysis: As an alternative to the above pipeline steps, it is possible to define a single, global method that directly calculates an enrichment p-value for a supplied data set and gene set. See `global` for the global analysis tests included in the package.

Several state-of-the-art analyses have predefined configuration objects in which the above steps are defined accordingly (see `predefinedAnalyses`).

Value

An object of class `gsAnalysis` with components corresponding to the above parameters.

See Also

`predefinedAnalyses`, `geneSetAnalysis`, `evaluateGeneSetUncertainty`, `gls`, `transformation`, `gss`, `global`, `significance`

Examples

```
data(exampleData)
# defines an analysis that corresponds to gsAna1()
gsa <- gsAnalysis(
name = "averageCorrelation",
gls = "gls.cor",
glsParameterNames = c("labs","method"),
transformation = "transformation.abs",
transformationParameterNames = NULL,
gss = "gss.mean",
gssParameterNames = NULL,
globalStat = NULL,
globalStatParameterNames = NULL,
significance = "significance.sampling",
significanceParameterNames = c("numSamples"),
testAlternative = "greater")
print(gsa)


  # apply the previously defined analysis
  res <- geneSetAnalysis(
   # global parameters
   dat = countdata,
   geneSets = pathways[1],
   analysis = gsa,
   # parameters for the specific analysis gsAna1
   labs = labels,
   numSamples = 10)
```

`hist` *Null distribution histogram and statistic of the input set for enrichment analyses.*

Description

Plots the distribution of gene set statistic values obtained in different resampling settings of an enrichment analysis, and draws the statistic value of the input set as a vertical line.

Usage

```
## S3 method for class 'gsaResult'
hist(x,
signLevel = x$signLevel,
subset = NULL,
ask = FALSE,
addLegend = TRUE,
...)
```

Arguments

x	A result of a call to geneSetAnalysis (see also Details).
signLevel	The significance level that should be applied for the plots. Default is the significance level used for the analysis in x.
subset	Indices for the results that should be included in the diagram.
ask	If set to true, the plot function will prompt for a user input for each new plot that is shown on an interactive device (see par("ask")).
addLegend	If set to true (default), a legend is added to the plot.
...	Other parameters which can be used for histograms (see hist).

Details

The function plots the distribution of gene set statistic values under the null hypothesis. It requires the significance assessment step of the enrichment analysis configuration (parameter significance or gsAnalysis) to be a computer-intensive testing procedure that yields a distribution of gene set statistic p-values under the null hypothesis. Predefined configurations for which this plot works are analysis.gsea, analysis.averageCorrelation and analysis.averageTStatistic.

A histogram is plotted for the analysis in x. If x includes the analyses for several gene sets, one histogram is plotted for each of the gene sets.

The statistic value of the input set is depicted as a vertical line.

The most common graphical parameters can be supplied as vectors (one entry per analyzed gene set) to vary them between the different analyses. These parameters are: main, xlab, ylab.

Value

Returns a list with all the underlying data for the plotted histograms as invisible object.

See Also

geneSetAnalysis, predefinedAnalyses, gsAnalysis, evaluateGeneSetUncertainty, plot.uncertaintyResult

Examples

```
# load data
data(exampleData)
  res <- geneSetAnalysis(
   # global parameters
   dat = countdata,
   geneSets = pathways[3],
   analysis = analysis.averageCorrelation(),
   # additional parameters for analysis.averageCorrelation
   labs = labels,
   p = 1,
   numSamples = 10)

  # plot the histogram for the cell cycle control gene set
  hist(res, main = names(pathways[3]))
```

labels — *Example data to label samples in countdata (rows) in two classes 0 and 1*

Description

Randomly generated example data to give samples in countdata a class label 0 or 1

Usage

```
data(exampleData)
```

Details

The data consists of a variable `labels` with a vector of 96 values. The values are a random sequence of 0's and 1's.

Examples

```
data(exampleData)

print(labels)
```

mergeProbesForGenes — *Merge multiple probes for one gene*

Description

Merges all probes belonging to the same gene by identifying duplicate row names in a data matrix.

Usage

```
mergeProbesForGenes(dat,
method = c("mean", "max", "min", "median"))
```

Arguments

`dat`	A numeric matrix of gene expression values for all analyzed genes. Here, each row corresponds to one probe, and each column corresponds to one sample. The rows must be named with the gene names and may contain duplicates if multiple probes correspond to the same gene.
`method`	The method which should be used to merge probes entries in `dat`. Depending on the chosen method, the merged value for a gene and a specific sample is defined as the mean value, the maximum value, the minimum value or the median of all probes of this sample belonging to the gene.

Value

A matrix of the same structure as `dat`, but possibly with fewer rows if probes were merged.

See Also

`geneSetAnalysis`

Examples

```
dat <- matrix(1:6, nrow=3, ncol=2)
rownames(dat) <- c("g1", "g2", "g1")

newDat <- mergeProbesForGenes(dat, method = "mean")
```

`parse GMT files` *Venn Euler Diagramm*

Description

Parses a GMT file as downloadable from MSigDB (presented in Subramanian et al.) and returns a list of gene sets.

Usage

```
parseGmt(file)
```

Arguments

`file`	A file name.

Details

Parses a GMT file and returns a list of gene sets. Each list element named according to the included gene set. The gene set files can be downloaded from http://www.broadinstitute.org/gsea/msigdb.

Value

A named `list` of gene sets.

References

Subramanian, A., Tamayo, P., Mootha, V. K., Mukherjee, S., Ebert, B. L., Gillette, M. A., Paulovich, A., Pomeroy, S. L., Golub, T. R., Lander, E. S., Mesirov, J. P. (2005) Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Science of the United States of America*, **102**, 15545–15550.

See Also

`geneSetAnalysis`, `predefinedAnalyses`, `gsAnalysis`

pathways *Example data for pathways to to gene set enrichtment analyses*

Description

A list comprising exemplary pathways. Each pathway is represented by a vectro of random genes from countdata which are associated to this pathway.

Usage

```
data(exampleData)
```

Details

The data consists of a variable `pathways` of class `list` with 9 exemplary pathways. Each element of the list corresponds to one pathway - represented by a vector of gene names which are involved in this pathway.

Examples

```
data(exampleData)

print(pathways)
```

plot *Plots the results of an uncertainty analysis.*

Description

For each percentage of original gene set genes, the quantiles of the distribution obtained by a re-sampling simulation are plotted. Significance threshold (quantile of the Null distribution) and the test statistic of the original gene set are drawn as horizontal lines.

Usage

```
## S3 method for class 'uncertaintyResult'
plot(x,
signLevel = x$signLevel,
addLegend = TRUE,
addMinimalStability = FALSE,
...)
```

Arguments

x	A result of a call to evaluateGeneSetUncertainty (see also Details).
signLevel	Only results with significance level smaller than the given value are plotted.
addLegend	If set to true (default), a legend is added to the plot.
addMinimalStability	If set to true, a line is added to the plot giving the minimal stability.
...	Other parameters which can be used for histograms (see plot).

Details

The function plots the quantiles of the resampling distributions for evaluated degrees of fuzziness. It requires the significance assessment step of the enrichment analysis configuration (parameter significance or gsAnalysis) to be a computer-intensive testing procedure that yields a distribution of gene set statistic values under the null hypothesis. Predefined configurations for which this plot works are analysis.gsea, analysis.averageCorrelation and analysis.averageTStatistic.

Three lines, corresponding to the different qunatiles with one dot per fuzziness evaluation (k) are plotted for the analysis in x. The significance threshold is shown as a green horizontal line. The statistic value of the original input set is depicted as a red horizontal line.

If addMinimalStability is TRUE, the lower bound of the stability is ploted as a dotted line.

Value

No return value, called for creating a plot.

See Also

geneSetAnalysis, predefinedAnalyses, gsAnalysis, evaluateGeneSetUncertainty

Examples

```
# load data
data(exampleData)

  res <- evaluateGeneSetUncertainty(
   # parameters for evaluateGeneSetUncertainty
   dat = countdata,
   geneSet = pathways[[1]],
   analysis = analysis.averageCorrelation(),
   numSamplesUncertainty = 10,
   N = seq(0.1,0.9, by=0.1),
   # additional parameters for analysis.averageCorrelation
   labs = labels,
   numSamples = 10)

  # plot the results for the cell cycle control gene set
  plot(res, addMinimalStability = TRUE)
```

plotOverrepresentation *Plot overlap of gene sets and core set*

Description

Plots a Venn diagramm of the overlaps of the core set and gene sets in an overrepresentation analysis.

Usage

```
plotOverrepresentation(
object,
signLevel = object$signLevel,
subset = NULL,
aggregate = FALSE,
ask = FALSE,
...)
```

Arguments

object	A result of a call to geneSetAnalysis using the predefined analysis analysis.customOverrepresentation or analysis.overrepresentation.
signLevel	Only results with significance level smaller than the given value are included in the venn diagram.
subset	Indices for the results that should be included in the diagram.
aggregate	Specifies whether all gene sets should be plotted in a single Venn diagram (which is possible for at most four gene sets) or whether there should be one Venn diagram for each gene set.
ask	If set to true, the plot function will prompt for a user input for each new plot that is shown on an interactive device (see par("ask")). If aggregate = TRUE, ask is ignored.
...	Further parameters to be passed to vennDiagram.

Value

No return value, called for creating a plot.

See Also

geneSetAnalysis, predefinedAnalyses, gsAnalysis

Examples

```
data(exampleData)
  # use the absolute correlation as a gene-level statistic
  stat <- abs(apply(countdata,1,cor,y = labels))
  # define the core set as the 25% genes with the highest correlation
  coreSet <- rownames(countdata)[tail(order(stat), 25)]

  # perform an overrepresentation analysis
  resOverrep <- geneSetAnalysis(
   dat = countdata,
   geneSets = pathways,
   analysis = analysis.customOverrepresentation(),
   coreSet = coreSet,
   adjustmentMethod = "fdr")

  # plot a Venn diagram
  plotOverrepresentation(resOverrep, subset = 1:3, aggregate = TRUE)
```

---

predefinedAnalyses *Predefined enrichment analyses*

Description

Predefined analysis configurations that can be used in geneSetAnalysis

Usage

```
analysis.gsea()
analysis.overrepresentation()
analysis.customOverrepresentation()
analysis.averageCorrelation()
analysis.averageTStatistic()
analysis.globalTest()
analysis.globalAncova()
```

Details

The above functions return configurations for state-of-the-art analysis pipelines that can be used in geneSetAnalysis. All configurations are preconfigured collections of standard methods for the different pipeline steps. The following lists the methods chosen for the different steps and their parameters. For more detailed descriptions of these methods, please refer to the linked manual pages.

- analysis.gsea defines the Gene Set Enrichment Analysis (GSEA) method by Subramanain et al. Here, the gene-level statistic the absolute correlation calculated by gls.cor with the associated parameters labs, method and a preprocessing by transformation.abs. As a gene set statistic, the enrichment score (function gss.enrichmentScore with parameter p) is calculated. The significance is assessed in a permutation test using significance.permutation with testAlternative = "greater" and free parameter numSamples, labs.

- `analysis.overrepresentation` calculates an overrepresentation analysis using the gene-level statistic `gls.tStatistic` with parameters `pValue`(should be `TRUE`), `alternative` and `labs`. The resulting values are then transformed via `transformation.adjustAndBinarize` (parameters are the `adjMethod` and `threshold`). Finally `gss.fisherExactTest` is used as gene set statistic.
- `analysis.customOverrepresentation` calculates an overrepresentation analysis using a user-defined core set `coreSet`. That is, instead of calculating this core set internally based on differential expression as the standard overrepresentation analysis, this function allows for defining custom core sets. It internally uses the global analysis `global.overrepresentation`.
- `analysis.averageCorrelation` calculates the gene-level statistic as the absolute correlation using `gls.cor` (with parameters `labs, method`) and `transformation.abs`. The gene set statistic is the mean correlation calculated by `gss.mean`. The significance is assessed by comparing the gene set statistic to randomly sampled gene sets using `significance.sampling` (with the parameter `numSamples` and the preset parameter `testAlternative = "greater"`).
- `analysis.averageTStatistic` uses the absolute t statistic as the gene-level statistic by applying `gls.tStatistic` (with parameters `labs, pValue, alternative`) and `transformation.abs`. The gene set statistic is the mean t statistic in the gene set as returned by `gss.mean`. The significance is assessed by comparing the gene set statistic to randomly sampled gene sets using `significance.sampling` (with the parameter `numSamples` and the preset parameter `testAlternative = "greater"`).
- `analysis.globalTest` performs a global gene set enrichment analysis by Goeman et al. by applying the `global.test` function which in turn wraps the `gt` function in the **globaltest** package.
- `analysis.globalAncova` applies the global ANCOVA method by Hummel et al. using the global method `global.ancova` which wraps the `GlobalAncova` function in the **GlobalAncova** package.

Value

All functions return an object of class `gsAnalysis` that specifies the corresponding analysis parameters for `geneSetAnalysis`.

References

Subramanian, A., Tamayo, P., Mootha, V. K., Mukherjee, S., Ebert, B. L., Gillette, M. A., Paulovich, A., Pomeroy, S. L., Golub, T. R., Lander, E. S., Mesirov, J. P. (2005) Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Science of the United States of America*, **102**, 15545–15550.

Hummel, M., Meister, R., Mansmann, U. (2008) GlobalANCOVA: exploration and assessment of gene group effects. *Bioinformatics*, **24**(1), 78–85.

Goeman, J. J., van de Geer, S. A., de Kort, F., van Houwelingen, H. C. (2004) A global test for groups of genes: testing association with a clinical outcome. *Bioinformatics*, **20**(1), 93–99.

See Also

`geneSetAnalysis`, `gsAnalysis`

Examples

```
data(exampleData)
  # apply a gene set analysis based on the average absolute correlation
  resAvCor <- geneSetAnalysis(
   # parameters for geneSetAnalysis
   dat = countdata,
   geneSets = pathways[1],
   analysis = analysis.averageCorrelation(),
   adjustmentMethod = "fdr",
   # additional parameters for analysis.averageCorrelation
    labs = labels,
   method = "pearson",
   numSamples = 10)

  # apply an overrepresentation analysis
  resOverrep <- geneSetAnalysis(
   # parameters for geneSetAnalysis
   dat = countdata,
   geneSets = pathways,
   analysis = analysis.overrepresentation(),
   adjustmentMethod = "fdr",
   # additional parameters for analysis.overrepresentation
   pValue = TRUE,
   threshold = 0.1,
   labs = labels
  )

  # apply a global analysis using GlobalAncova
  resGA <- geneSetAnalysis(
   # parameters for geneSetAnalysis
   dat = countdata,
   geneSets = pathways[1],
   analysis = analysis.globalAncova(),
   adjustmentMethod = "fdr",
   # additional parameters for analysis.globalAncova
   labs = labels,
   method = "approx")
```

`preprocessGeneSets` *Eliminate unknown genes from gene sets*

Description

This function removes all genes that are not part of the experiment (not in `rownames(dat)`) from the specified gene sets which. All names are set to lower case.

Usage

```
preprocessGs(
dat,
geneSets)
```

Arguments

dat	A numeric matrix of gene expression values for all analyzed genes. Here, each row corresponds to one gene, and each column corresponds to one sample. The rows must be named with the gene names used in the gene sets. Here, only the row names (i.e. the gene names) are used by the function.
geneSets	A list of gene sets to be processed, where each gene set is a vector of gene names corresponding to the row names of dat.

Value

A list of preprocessed gene sets, where each gene set only contains those genes that are also present in dat

See Also

geneSetAnalysis

Examples

```
#values are not important, only the row names are used
dat <- matrix(0,100,10)
rownames(dat) <- paste("gene",1:100,sep="")

geneSets <- list(
gs1 = paste("GENE",1:20,sep=""),# all genes in the analyzed data
gs2 = paste("Gene",101:110,sep=""),#no gene in the analyzed data
gs3 = paste("gene",90:110,sep="")#some genes in the analyzed data
)

newGeneSets <- preprocessGs(dat = dat, geneSets = geneSets)
```

SignificanceAssessment

Significance assessment

Description

Functions to assess the significance of the gene-level statistics, as used in the significance parameter of gsAnalysis. These functions are based on applying the same analysis to randomly modified data sets or gene sets and comparing their statistic values to the original gene set statistic value.

Usage

```
significance.sampling(
...,
dat,
geneSet,
analysis,
glsValues,
numSamples = 1000)

significance.permutation(
...,
dat,
geneSet,
analysis,
glsValues,
numSamples = 1000,
labs)

significance.restandardization(
...,
dat,
geneSet,
analysis,
glsValues,
numSamples = 1000,
labs)
```

Arguments

`...`	Additional parameters for the different steps of the analysis pipeline, depending on the concrete configuration supplied in `analysis`.
`dat`	The original data set as a numeric matrix of gene expression values for all analyzed genes. Here, each row corresponds to one gene, and each column corresponds to one sample. The rows must be named with the gene names used in the gene sets.
`geneSet`	The original gene set in form of a vector of gene names corresponding to the row names of `dat`.
`analysis`	The analysis applied to the original gene set (that should also be applied to the modified gene sets). This is an object of type `gsAnalysis` as produced by the function `gsAnalysis`.
`glsValues`	A vector containing the (possibly transformed) gene-level statistic values for each gene in the original data set `dat`.
`numSamples`	The number of random samples that should be taked to calculate the null distribution for the significance assessment. Default is `1000` for each test.
`labs`	A vector of class labels for the samples in `dat` for `significance.permutation` and `significance.restandardization`.

Details

Standard methods for the significance assessment of a gene set statistic (to be used in an analysis pipeline defined by `gsAnalysis`):

- `significance.sampling`: This function repeatedly draws random gene sets. Their gene set statistic values form the null distribution.
- `significance.permutation`: This function repeatedly permutes the labels of the data set. The gene set statistic values for the original gene set on the permuted data set form the null distribution.
- `significance.restandardization`: This function applies both a gene set sampling and a label permutation. The permutation statistic values are standardized by their mean and standard deviation and then restandardized based on the gene set sampling statistic values. These restandardized values form the null distribution (Efron and Tibshirani).

Value

`significance.sampling` returns a list with the following elements:

`gssValues`	A vector of gene set statistic values, one entry per sample.
`randomGeneSets`	A matrix containing the gene sets which were sampled randomly from the set of all genes.

`significance.permutation` returns a list with the following elements:

`gssValues`	A vector of gene set statistics, one entry per sample.
`permutations`	A matrix, where each column contains the indices of one permutation.

`significance.restandardization` returns a list with the following elements:

`gssValues`	A vector of gene set statistics, one entry per sample.
`samplingValues`	A list of sub-lists, each containing one sampling result as defined above.
`permutationValues`	A list of sub-lists, each containing one permutation result as defined above.

References

Efron, B., Tibshirani, R. (2007) On testing the significance of sets of genes. *Annals of Applied Statistics*, **1**, 107-129.

See Also

`geneSetAnalysis`, `gsAnalysis`, `hist.gsaResult`

summary.gsaResult *Summarize gene set analysis results*

Description

Prints a summary of a gene set analysis result object.

Usage

```
## S3 method for class 'gsaResult'
summary(object,
mode = c("summary", "table"),
orderBy = c("adjustedPValues", "rawPValues", "geneSetName"),
significantOnly = FALSE,
signLevel = object$signLevel,
...)
```

Arguments

object	A result object as returned by geneSetAnalysis.
mode	Specifies the type of information that is displayed: By default (mode="summary"), a brief summary of the number of significant and insignificant gene sets is printed. For mode="table", createSummaryTable is called, and a detailed table of adjusted and unadjusted p-values and the number of genes for each gene set is printed.
orderBy	If mode="table", this specifies which field should be used for the row ordering. By default, rows are ordered according to the adjusted p-values.
significantOnly	If mode="table", this specifies whether all gene sets (significantOnly=FALSE) or only the statistically significant gene sets (significantOnly=TRUE) should be included in the table.
signLevel	If mode="table" and significantOnly=TRUE, this specifies the significance level for the results that should be included in the table. By default, the original significance level of the analysis is used.
...	Currently unused

Value

Returns a data.frame with a summary of the produced results such as data set name, p-values, size of the given gene set.

See Also

geneSetAnalysis, hist.gsaResult, createSummaryTable

Examples

```
data(exampleData)
  # perform gene set analyses for several pathways
  res <- geneSetAnalysis(
   # global parameters
   dat = countdata,
   geneSets = pathways,
   analysis = analysis.averageCorrelation(),
   # additional parameters for analysis.averageCorrelation
   labs = labels,
   numSamples = 10)

  #summarize the analyses
  summary(res, mode = "summary")

  summary(res, mode = "table", orderBy = "rawPValues")
```

`Transformations` *Transformations*

Description

Functions to transform the gene-level statistic values prior to the calculation of the gene set statistics, as used in the `transformation` parameter of `gsAnalysis`. Most of the functions wrap existing R functions.

Usage

```
transformation.abs(x)

transformation.square(x)

transformation.localFdr(x,
statistic="pvalue",
cutoff.method="fndr",
pct0=0.75)

transformation.binarize(x, quant)

transformation.rank(x)

transformation.adjust(x, adjMethod = "fdr")

transformation.adjustAndBinarize(x, adjMethod = "fdr", threshold = 0.05)
```

Arguments

x	A numeric vector of gene-level statistic values, one per gene. These values are calculated by the previous step (see gls).
statistic	Specifies the null model for transformation.localFdr (see statistic parameter of fdrtool for possible values).
cutoff.method	Type of cut-off method used in transformation.localFdr (see cutoff.method parameter of fdrtool for possible values).
pct0	Fraction of data used by transformation.localFdr if cutoff.method="pct0" (see fdrtool for a detailed description).
quant	For transformation.binarize, this numeric value in the interval [0,1] defines the percentage of gene-level statistic values which should be set to zero. The remaining values are set to one.
adjMethod	The method to use for the adjustment for multiple testing (see method parameter of p.adjust for possible values).
threshold	The threshold for differential expression of a gene (defaults to 0.05). Values smaller than these threshold are set to 1 , others to 0.

Details

Standard transformation functions for gene-level statistics (to be used in an analysis pipeline defined by gsAnalysis):

- transformation.abs: Calculates the absolute values of the elements in x (a wrapper for abs).
- transformation.square: Squares all elements in x.
- transformation.localFdr: Calculates the local fdr for the elements in x. This is a wrapper for fdrtool.
- transformation.binarize: Binarizes the values in x by using the quant quantile as a threshold.
- transformation.rank: Ranks the values in x and returns the rank vector.
- transformation.adjust: Adjusts for multiple testing according to the adjustment method specified in adjMethod.
- transformation.adjustAndBinarize: Adjusts for multiple testing according to the adjustment method specified in adjMethod and binarizes the resulting p-values according to threshold (values smaller than the threshold become 1 others 0).

Value

All functions return a vector of transformed values having the same length as x.

See Also

geneSetAnalysis, gsAnalysis, gss, gls

Index